N89-26593

Spacelab Life Sciences-1 Electrical Diagnostic Expert System*

C. Y. Kao, W. S. Morris

General Electric Government Services
Mission Integration Office
1050 Bay Area Blvd.
Houston, Texas 77058 (713)488-9005

ABSTRACT

The Spacelab Life Sciences-1 (SLS-1) Electrical Diagnostic (SLED) expert system is a continuous, real time knowledge-based system to monitor and diagnose electrical system problems in the Spacelab. After fault isolation, the SLED system provides corrective procedures and advice to the ground-based console operator.

The SLED system uses the Unit (frame) of KEE to represent the knowledge about the electrical components and uses KEE-Bitmaps to represent the electrical schematics. The diagnostic logic, stored as a set of LISP structure, mirrors that in the malfunction procedures defined in the Spacelab Flight Data File Malfunction Procedures Handbook (JSC-18927) of NASA. The SLED system utilizes downlink telemetry data as input. The system performs some initial screening of the data in order to recognize patterns representing serious problems, and updates its knowledge about the status of Spacelab every 3 seconds. Important parameters are monitored via Active-Values within KEE. The Active-Value kicks off the diagnostic analysis to determine the source of the problems if any problem has been identified. The system supports multiprocessing of malfunctions and allows multiple failures to be handled simultaneously. The user can examine each of the reported problems and receive corrective advice related to each problem. Information which is readily available via a mouse click includes: general information about the system and each component, the electrical schematics, the recovery procedures of each malfunction, and an explanation of the diagnosis.

A rich set of user interfaces is provided in SLED. Various tools have been included which allow a non-programmer to define new diagnostic procedures, define and update schematics of the electrical system, and to change the SLED model by changing the graphical representation. Each tool and function has explanatory prompts to aid the user.

* **This work is performed by GE Government Services, Johnson Space Center, Houston Texas in support of the NASA Mission Management Office, Mission Manager D. Womack. Government Contract NAS9-17884**

1. INTRODUCTION

The Spacelab Life Sciences-1 (SLS-1) Experiment Electrical Diagnostic (*SLED*) system is a continuous, real time knowledge-based system to monitor and diagnose experiment equipment electrical system problems in Spacelab. After fault isolation, the *SLED* system provides corrective procedures and advice to the ground-based console operator. Operation of the system is to be continuous throughout the SLS-1 mission. The inputs to *SLED* are a stream of parameters downlinked from the Tracking and Data Relay Satellite (TDRS) system. This paper covers the functionality of the *SLED* system , and compares it with the current conventional approach for diagnosing experiment electrical problems in Spacelab.

Subsequent sections to follow will include a background description on how the *SLED* system evolved, the application domain characteristics and issues, the system design overview, the current status, and the conclusion comments.

Background

The console operation of the Payload Operations Control Center (POCC) is an important but tedious task. Some of the Avionics Systems Payload Systems Engineer (PSE)'s tasks involve trouble shooting and determining the status of hardware between the Spacelab/Experiment/Mission Peculiar Equipment (MPE) Systems. They also assist in resolution of hardware problems with Johnson Space Center (JSC) Mission Control Center (MCC), Principal Investigator (PI) teams, Science Monitoring Area (SMA), and POCC Cadre positions.

The current approach for the detection and resolution of Spacelab experiment electrical failures begins with the PSE observing console displayed data for out-of-limit errors. Once an error is detected, the PSE determines what type of error has occurred and searches through the Flight Data File Spacelab Malfunction Procedures, [JSC-18927], to find, isolate, and provide recovery procedures for the electrical failure. The procedure that the PSE follows includes looking at other consol displayed parameters to determine the status of interrelated hardware. The PSE then looks at the schematic for the system, which is found in another handbook, S/L Systems Handbook, [JSC-12777C]. Also, the procedure may require communicating with the crew to perform some type of activity in the Spacelab Module, which may be necessary to diagnose the failure correctly. Having isolated the failure, the malfunction procedure includes or identifies a procedure (Sub System Recovery (SSR)) for recovery to a nominal configuration. These recovery procedures list what actions are to be taken by the crew, the equipment that has been lost due to the failure, crew indications, and additional notes. The SSR procedures may include communicating with the crew/POCC to perform some type of action in order to resolve the failure. The task is very time consuming and can produce many human errors. This is the flow that the crewmember/PSE has to perform for every malfunction that occurs. If multiple errors occur at one time this not only adds to the confusion for the PSE, but also introduces a probability of more human errors.

The *SLED* system was devised from a General Diagnostic system, which was initiated in late 1986, to aid the tedious task of diagnosing Spacelab failures. This system's main purpose is to make the console operation task of the

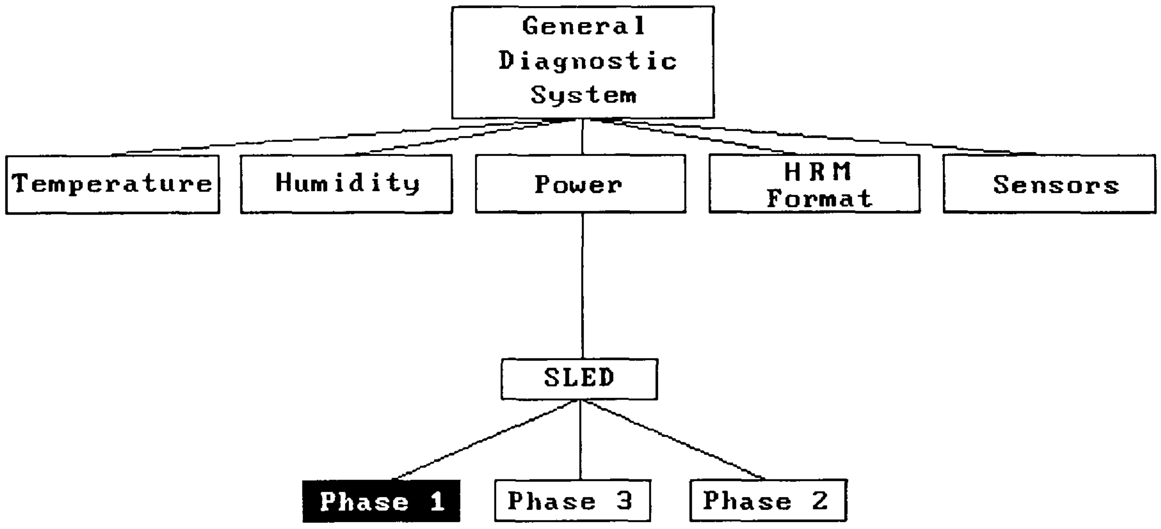


Figure 1. How the current SLED system originated.

PSE easier. Its capabilities were to include diagnostics concerning power, temperature, the high rate multiplexer (HRM) format, humidity, and sensors (Figure 1). Each of these areas constitute comprehensive malfunction recovery procedures. In other words, the intended expert system will be able to diagnose power problems, high and low temperature problems, HRM format irregularities, problems related to the high and low humidity and problems resulting from bad sensors. Designing an expert system having all these diagnostics to work in a real-time useful manner is a very large task. After spending more then half a year in prototyping, the initiators decided to break the large problem into smaller problems. This is how the *SLED* system evolved, from the General Diagnostic system. As with each of the other subdomains, temperature, humidity, HRM format, and sensors, can be their own expert system. Integrated together as a whole, they can interact as one large domain expert system. The *SLED* system was initially initiated to include the entire electrical system from the electrical power distribution system (EPDS) up to and including the experiment hardware. This system was still a large problem. The problem was simplified by splitting it into three phases. Phase one implements the EPDS, phase two covers the hardware beginning where the EPDS left off up to the "box" that contains the experiment hardware, and finally, the third phase covers the experiment hardware. Phase one is completed and is described in detail in this paper.

Objective of SLED

The objective of the *SLED* system is to develop a continuous, real-time expert system to monitor and diagnose the electrical power problems in the Spacelab and the experiment equipment. The primary goal is to help the PSE to monitor the telemetry data and to diagnose the malfunction by firing the diagnostic procedure automatically when an out-of-limit condition is detected, especially in the event of multiple-failures when the PSE is most confused by the abnormal telemetry data. The secondary purpose of the *SLED* system is to aid the PSE in fixing these problems by calling up the malfunction procedure and the SSR procedure. Because we have covered all of the malfunctions that appear in the EPDS schematics, we anticipate that the *SLED* system will find the causes of most of the system faults. In the other cases when the *SLED* system failed to identify the problem or failed to give correct malfunction procedures, the system should jog the PSE's memory enough to solve the problem with the help of *SLED*'s schematic display facility and the available status information of every component in the schematics.

2. APPLICATION DOMAIN and ISSUES

The basic Spacelab EPDS distributes DC-Main, DC-Essential and 400 Hz AC power to Spacelab experiment equipment and also provides the necessary power to Spacelab subsystems. The EPDS receives its DC power (+28V nominal) from Orbiter hydrogen/oxygen fuel cells through the Orbiter bus system which, in turn, is connected to the Spacelab Power Control Box (PCB) and the Spacelab Emergency Box. The AC power is generated from the DC main power by the Spacelab 400 Hz inverters. Dedicated Spacelab Subsystem (S/S) and experiment DC/AC inverters, which receive their primary power from the main DC power bus via the PCB, supply three-phase AC power to the S/S equipment and experiment equipment. For experiment caution and warning sensors, safing command actuators and critical experiment equipment needing power during all mission phases, a dedicated experiment essential power bus is routed through the Spacelab Module. This power bus receives its essential power from the Spacelab Emergency Box, which, in turn, is powered by the Orbiter auxiliary DC power busses.

Using Expert System technology in Space Mission Operations is a focus of current research in recent years since the promising benefits of the technology and the strong desire of automating the mission planning and mission control tasks to keep the Aerospace exploration cost down. [Dickey & Toussaint, 1984] describe a prototype expert system for the integration of housekeeping subsystems on board a manned space station. [Silverman, 1986] talks about a distributed expert system testbed for Spacecraft ground facilities. [Rook & Odubiyi, 1986] describes a prototype expert system to provide real-time aid/advice for ground based satellite orbit control operations. [Muratore, Madison & etc. 1988] describes the development of the real-time expert system prototype for shuttle Mission Control Center as a five layer model to integrate various monitoring and analysis technologies such as digital filtering, fault detection algorithms, and expert systems. [Hamilton, 1986] describes a prototype expert system application to detect and diagnose faults in attitude control systems. Besides these listed, there is an annual conference sponsored by NASA, Goddard Conference on Space Applications of Artificial Intelligence, which focuses on Artificial Intelligence in Space. Several papers presented in this conference are relevant to our target application domain of developing a monitoring and diagnostic expert system for Spacelab Power Subsystems that is capable of supporting real-time operation and diagnosing multiple faults. In particular, [Wilkinson, Happell & etc., 1988] described a prototype fault isolation expert system for TDRSS application in a real-time on-line environment, which is similar in nature to our system.

Three special characteristics of the application domain is important to the design and development of the expert system in space application.

1. **The domain is not mature and there is not a so called "expert" in the domain.** This implies that the system design should be flexible and the designer/implementer should be willing to change the system architecture if needed.

2. **The resulting expert system should be a continuous, real-time system, which is admitted to be difficult to achieve within current expert system technology.** As [D'Ambrusio, Fehling & etc., 1987] pointed out, few knowledge base systems have been developed for real-time applications. How to meet the real-time requirements in an expert system is an important issue. [Griesmer, Hong & etc., 1984] reported the implementation of the YES/MVS, which is a continuous real-time expert system, they also outlined the real- time requirements for expert systems. Specifically, they discuss how to modify OPS5 ([Forgy, 1981]) in support for real-time tasks namely: 1. Speed considerations; 2. Initialize an action like production firing at a given time; 3. Fast communications between modules; 4. Need for explicit control (by the operator); and 5. Some specific requirements of continuous operations. These continuous operation requirements includes: (a). Inference engine should not terminate; (b). Automatic restart capability; (c). Remove "garbage" work-memory element. Similar requirements for expert systems in space applications has been outlined in [Leinweber, 1987] namely: 1. High-Speed context-sensitive rule activation; 2. Efficient recycling of no-longer-needed memory elements; 3. Interactively accept command sequences from operators; 4. Fast communications between multiple expert systems. These real-time requirements affect our design decision. [Wilkinson, Happell, & etc., 1988] discuss the "hard" real-time requirements, which is "provide outputs by some deadline time". They also pointed out that in an AI system, the time taken to complete a calculation is nondeterministic . The state-of-the-art for such problems are often to build something to see if it will work in real time. Performance problems are solved via the bigger hammer theory: if it is not fast enough, buy a faster machine. They call this a "soft" real-time system. We are in the same boat, we plan to deliver the SLED system as a "soft" real-time system. We will discuss these issues later.

3. **This system should be able to detect and diagnose multiple faults, which is a tough problem too.** As we all know, fault diagnostic systems is one of the traditional expert system application areas,[Hayes-Roth, Waterman & Lenat, 1983]. For example, MYCIN ([Buchanan & Shortliffe, 1984]) is a classical example of a medical diagnostic system.

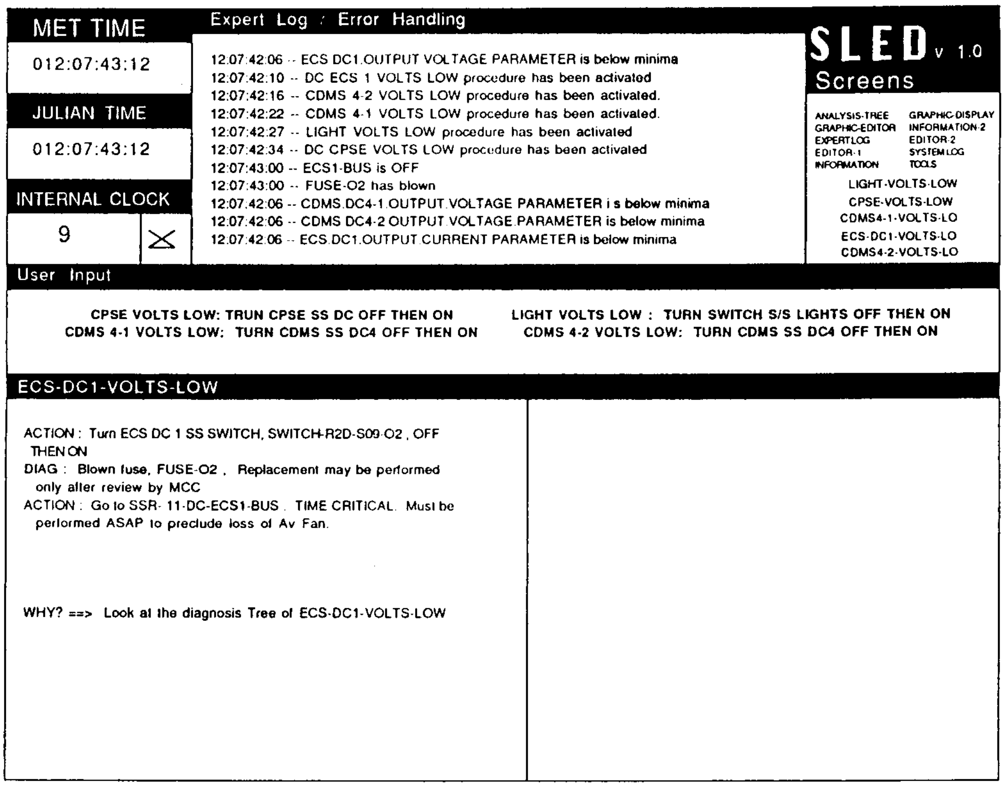
Instead of using shallow knowledge in electrical trouble shooting, it is argued that a design-model-based approach, rather than the traditional empirical-rule-based approach is necessary for electronic diagnostic systems. In this approach, the only available information is the system description, i.e. its design and structure, together with some observations of the system's behavior and a statistical characterization for the type of failure. [Davis, 1983] argued about using the first principle and structure knowledge in electronic trouble-shooting. [Milne, 1987] designed a system using "second principles", which are rules that an electronic engineer would use when diagnosing a circuit during a trouble-shooting task. [Cantone, Pipitone & etc., 1983] proposed the model-based probabilistic reasoning for electronic trouble-shooting. [Pipitone, 1984] used the qualitative causal model and generic component knowledge to produce a model for diagnostic reasoning. [Taie & Srihari, 1986] proposed a hierarchical device representation scheme using instantiation rules and structural template in a semantic network with procedural attachment for function description of a device. [Maletz, 1985] used context graphs combined with electrical system structures, diagnostic tests, and symptom knowledge to support multiple fault hypothesis and reasoning about the abilities of testing to discriminate among a collection of possible faults. [Geffner & Pearl, 1987] used belief networks to represent causal knowledge of system behavior, and using Bayesian inference for doing diagnosis. The distributed scheme then uses the independices embedded in a system to decompose the task of diagnosing the overall system into smaller sub-tasks of diagnosis for subparts of the net, then combined them together. Geffner & Pearl claim that the decomposition yields a globally-optimum diagnosis by local and concurrent computation using message- passing algorithms and attaining linear time in single-connected networks.

3. SYSTEM DESIGN OVERVIEW

The *SLED* system has been developed using the Texas Instruments (TI) Explorer workstation and the Knowledge Engineering Environment (KEE) expert system building tool ([KEE 2.1]). The *SLED* system uses the Unit (frame) of KEE to represent the knowledge about the electrical components and uses KEE-Bitmaps to represent the electrical schematics. When the downlink data is received, the values are put in their appropriate Unit within KEE. Utilizing the Active-Values of KEE, malfunction recovery procedures are activated. The decision for the activation of the recovery procedure is based upon a maximum and minimum value for that parameter.

SLED Approach

The *SLED* system approach for the detection and resolution of Spacelab experiment electrical failures incorporate system monitoring of the downlinked parameters for out-of-limit conditions. The software is designed to standby when no abnormal event occurs. The user will have no interaction with the system besides watching the telemetry data displays if he likes. When the system receives an out-of-limit condition a malfunction recovery procedure is activated. Once the procedure is activated a unique window is created for that malfunction. The diagnostic procedure may obtain any other value it needs in order to diagnose the out-of-limit condition. It may also request crew actions to be performed before the diagnosis can continue. The action that the crew is to take is displayed in the User Input window. Once the action has been completed the user would click on the displayed action and the recovery procedure resumes execution. When the system wants to let the user know something about the diagnosis, it displays this information in the unique window that was created upon activation.



```
12:07:42:06 -- ECS DC1.OUTPUT VOLTAGE PARAMETER is below minima
12:07:42:10 -- DC ECS 1 VOLTS LOW procedure has been activated
12:07:42:16 -- CDMS 4-2 VOLTS LOW procedure has been activated.
12:07:42:22 -- CDMS 4-1 VOLTS LOW procedure has been activated.
12:07:42:27 -- LIGHT VOLTS LOW procedure has been activated
12:07:42:34 -- DC CPSE VOLTS LOW procedure has been activated
12:07:43:00 -- ECS1-BUS is OFF
12:07:43:00 -- FUSE-O2 has blown
12:07:42:06 -- CDMS.DC4-1.OUTPUT.VOLTAGE PARAMETER i s below minima
12:07:42:06 -- CDMS DC4-2 OUTPUT VOLTAGE.PARAMETER is below minima
12:07:42:06 -- ECS.DC1.OUTPUT.CURRENT PARAMETER is below minima
```

```
CPSE VOLTS LOW: TRUN CPSE SS DC OFF THEN ON        LIGHT VOLTS LOW :  TURN SWITCH S/S LIGHTS OFF THEN ON
CDMS 4-1 VOLTS LOW:  TURN CDMS SS DC4 OFF THEN ON    CDMS 4-2 VOLTS LOW:  TURN CDMS SS DC4 OFF THEN ON
```

```
ACTION : Turn ECS DC 1 SS SWITCH, SWITCH-R2D-S09-O2 , OFF
 THEN ON
DIAG :  Blown fuse, FUSE-O2 ,  Replacement may be performed
  only after review by MCC
ACTION :  Go to SSR- 11-DC-ECS1-BUS .  TIME CRITICAL.  Must be
  performed ASAP to preclude loss of Av Fan.


WHY? ==>  Look at the diagnosis Tree of ECS-DC1-VOLTS-LOW
```

Figure 2. Malfunctions Window Display

Some of the text may be mousable so that the user may select it. Once the system has finished its diagnoses, the user will be notified. The mousable text that may be included in the text displayed by the system will allow the user to obtain additional information uniquely related to the out-of-limit condition being diagnosed. The type of additional information includes schematics of the related electrical components, specific information on an electrical component, SSR procedures, and a tree display showing how the system obtained its diagnosis. With this approach the user does not have to search through any manuals. All information is readily available. The system can detect all out-of-limit conditions that are pre-defined in the system. It is also capable of handling more than one out-of-limit condition, and can diagnose them simultaneously. (Figure 2)

Meet Real-Time Requirements

For supporting continuous operations, the main process of the *SLED* system will run forever. To protect it from aborting, we bounded the "Expert Log" window to be the terminal I/O and query I/O window while the *SLED* window frame is initially displayed. These bindings cause the Lisp machine to use the "Expert Log" window in error handling. If a system bug was ever to occur, and the user does not want to handle it, he can simply press the Abort-key to re-start the *SLED* system. We also put re-start capability in the process for data acquisition. Therefore, if any data connection problems occur, the two synchronized processes in (the simulation of) the Data Acquisition Layer will attempt to re-connect the data link, which keeps the data acquisition process running continuously.

One drawback in the KEE expert system shell is that the rule system is very slow. In order to meet the high-speed requirement of a real-time system we avoided using the rule system of KEE for the fault diagnostic process. Instead, we defined a frame-type Lisp structure to explicitly represent the diagnostic procedures. We also included a Lisp function to execute the diagnostic procedure defined in the frame. This way, the diagnostic reasoning is just a data-driven traversal of the malfunction specific analysis-tree. The execution of the diagnostic logic is tremendously fast. In fact, the time taken by the malfunction window, that is being displayed, and the user input is the only significant time consuming portion in the fault-diagnostic process. Of course, the decision of explicitly representing the diagnostic procedure may have some drawbacks. But the simplicity of the structure of each malfunction diagnostic procedure described in the Spacelab Malfunction Procedure Handbook makes this approach feasible. If the performance of the rule-system within KEE does not dramatically improve, and we are faced with more complicated diagnostic procedures, in order to meet the fast-speed real-time requirements, we may need to use other rule- engines, e.g. OPS5 ([Forgy, 1981]).

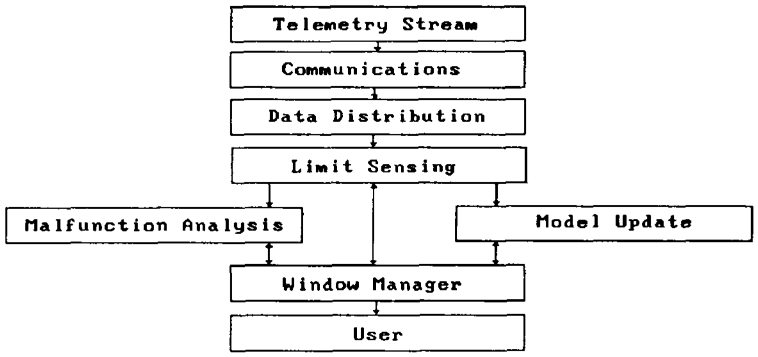
The architecture of Lisp-machines and the availability of some system software tools, in particular the stack group, the scheduler, the process structure, and the signal mechanism, make it easy to handle the real time requirements of initializing an action in a specific time, and other interrupt problems too. The requirement of high-speed context-sensitive activation of a procedure instead of rule activation in our case, is accomplished by using KEE's Active Value, which meets our timing requirements. As for the garbage collection issue, the fact that we do not use rules at all makes the problem of recycling of no-longer-needed memory elements (of the rule system) non-existent in our case. On the other hand, we utilize the resource facility to recycle our window objects and process objects and use list surgery, if possible, to keep down the rate of generating garbage. It seems that the *SLED* system does not get deteriorated by the Lisp-machine's garbage collection activity.

We depend on the mouse and menu system for explicit interactive operator control. Actually, the normal operator control functions are organized in the "Tools" window. By mouse clicking on an entry of the "Tools" window, the operator can interactively input stored command sequences to the *SLED* system. This organization is made possible by the fact that the mouse process of the Lisp-machine has a higher priority than other processes. Therefore, the system scheduler gives priority to the mouse process, hence it embeds user defined functions for mouse clicks.

Three Layer System Design

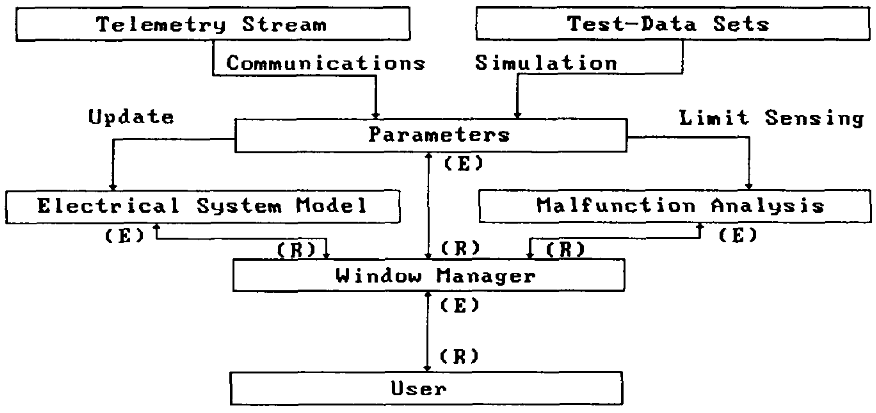
The parameters currently monitored by the *SLED* system are the parameters that appear in the DC distribution drawing and the AC distribution drawing of the Spacelab Systems Handbook. They include: parameters that trigger malfunction analysis procedures, parameters that are referenced by a malfunction analysis procedure, and parameters that indicate an isolated failure.

Object-oriented programming paradigm is used to simplify the design and the implementation of *SLED*. The major functional blocks of the diagnostic task have been turned into objects. These objects model the real world ele-



Down arrows show how data flows during normal operation.
Up arrows indicate re-configuration capabilities.

Figure 3. Functional Blocks Diagram.



R: Down arrow shows the Reporting in normal operation
E: Up arrow shows the Editing path in re-configuration

Figure 4. Object Blocks Diagram.

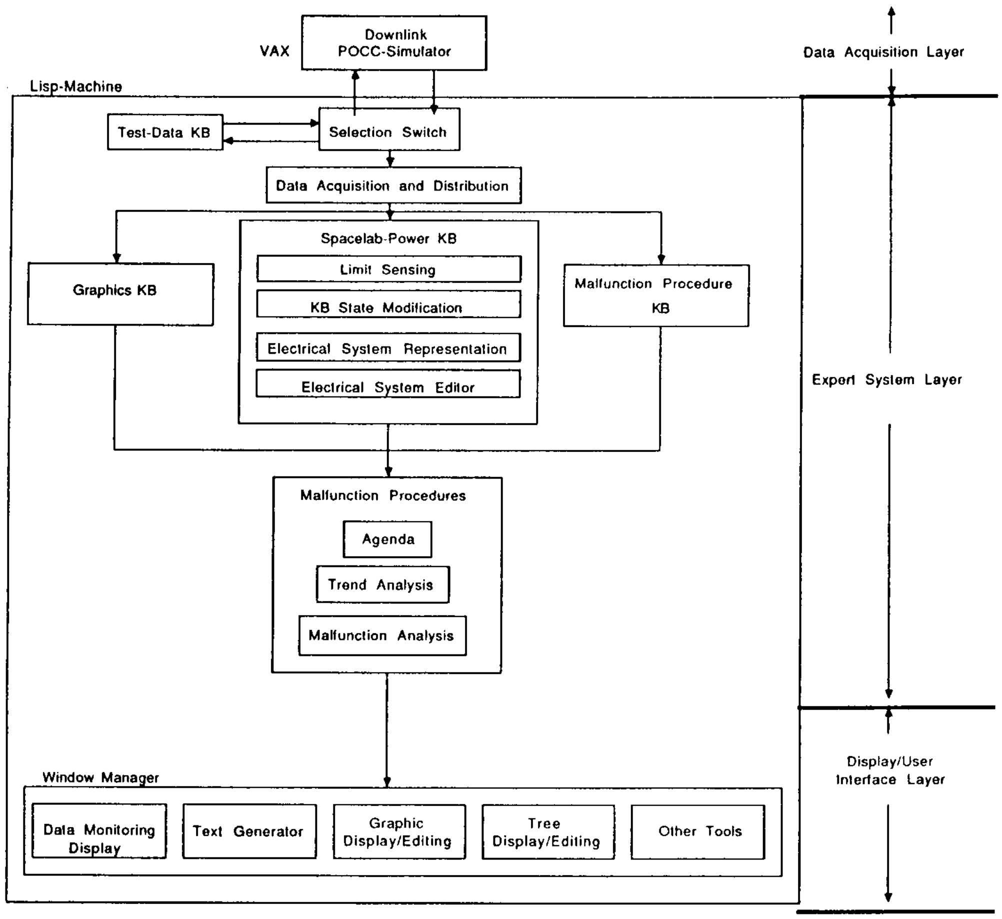


Figure 5. Three Layer System Design

ments which are relevant to the problems to be solved. Interfaces between functional blocks are thus limited to a few interactions in the form of message passing. The control structure is thus decentralized as an outcome of the object-oriented programming practice. Knowledge has been partitioned in different knowledge bases, which also adds a degree of specialization to all the functions included in every KB. A functional block diagram (Figure 3) and an object block diagram (Figure 4) of the target system help explain the structure of the *SLED* software.

As pointed out in [Strandberg, Abramovich, & etc., 1985], the layered-structure view offers a good organization for discussing system design. We can view the *SLED* system as a three layer structure (Figure 5). The first layer is the Data Acquisition Layer, the second is the Expert System Layer, and the third is the Display/User Interface Layer.

Data Acquisition Layer

As described in [LS-50044-A] and [SLP/2104], the Spacelab scientific data is routed through the subsystem and experiment I/O units, the 192 kb/s telemetry channel, composed of Orbiter and Spacelab data, is available and split up into: two voice channels at 32 kb/s, Orbiter-telemetry data at 64 kb/s nominal, and Spacelab data from experiment and subsystem I/O unit outputs at 64 kb/s nominal. This telemetry channel is software controlled through the Pulse Code Modulation Master Unit (PCMMU). It acquires the data from different sources (Orbiter General Processing Computer, subsystem I/O, and experiment I/O) in a demand and response manner. The Orbiter telemetry data will not need 64 kb/s all the time, so it might be possible that the subsystem and experiment data can be transmitted at a higher rate than 64 kb/s via this telemetry channel. The PCMMU can request data from the Command and Data Management System (CDMS) computers up to 2000 times per second. Upon one of these requests up to 10 data words can be transferred.

Controlled by the Network Signal Processor from the PCMMU, the 192 kb/s telemetry channel is transmitted to ground either via Space Tracking and Data Network (STDN) to the appropriate STDN ground station or via Tracking and Data Relay Satellite System (TDRSS) Ku-Band to the TDRSS ground station.

The Ku-Band data stream is down linked to White Sands, where the data are relayed to the JSC Mission Control Center (MCC). The digital data are recorded and at the same time being fed to a High Rate Demultiplexer (HRDM) in the Payload Operations Control Center (POCC). From the HRDM, real-time data streams are directly fed into the POCC Data Select Switch (PDSS) for distribution.

The data acquisition layer will be resident in the POCC's VAX computer to distribute the synchronized telemetry data to the Expert System Layer resident in the Lisp-Machine via DECnet connection every 3 seconds. Currently, we have not implemented the Data Acquisition Layer. Instead, we are using a set of hand- crafted test-data cases located in the Test-Data KB, along with two synchronized processes: data acquisition process and run sequence process in the LISP-Machine to simulate the Data-Acquisition Layer.

Expert System Layer

The Expert System Layer consists of the KBs and the inference mechanism of using the KBs. We are utilizing IntelliCorp's expert system shell, Knowledge Engineering Environment (KEE), to implement this layer.

There are four KB's in the *SLED* system: the Spacelab-Power KB, the Graphics KB, the Procedures KB, and the Test-Data KB. The Graphics KB is simply filled with run-time information for the application to perform some graphical editing and graphical display operations to support the Display/User Interface Layer. The Procedures KB is used to store the diagnostic text that is displayed in the Malfunction Window and the SSR text. The Test- Data KB is used to store the hand- crafted test data to simulate the Data-Acquisition Layer. This KB is temporary in nature and will be eliminated when the POCC Simulator or the Data-Acquisition Layer is available for generating testing data.

The Spacelab-Power KB is the main KB in the *SLED* system. It changes constantly. It contains the input parameters, the electrical system representation, and some data for the control structure. The representation is based

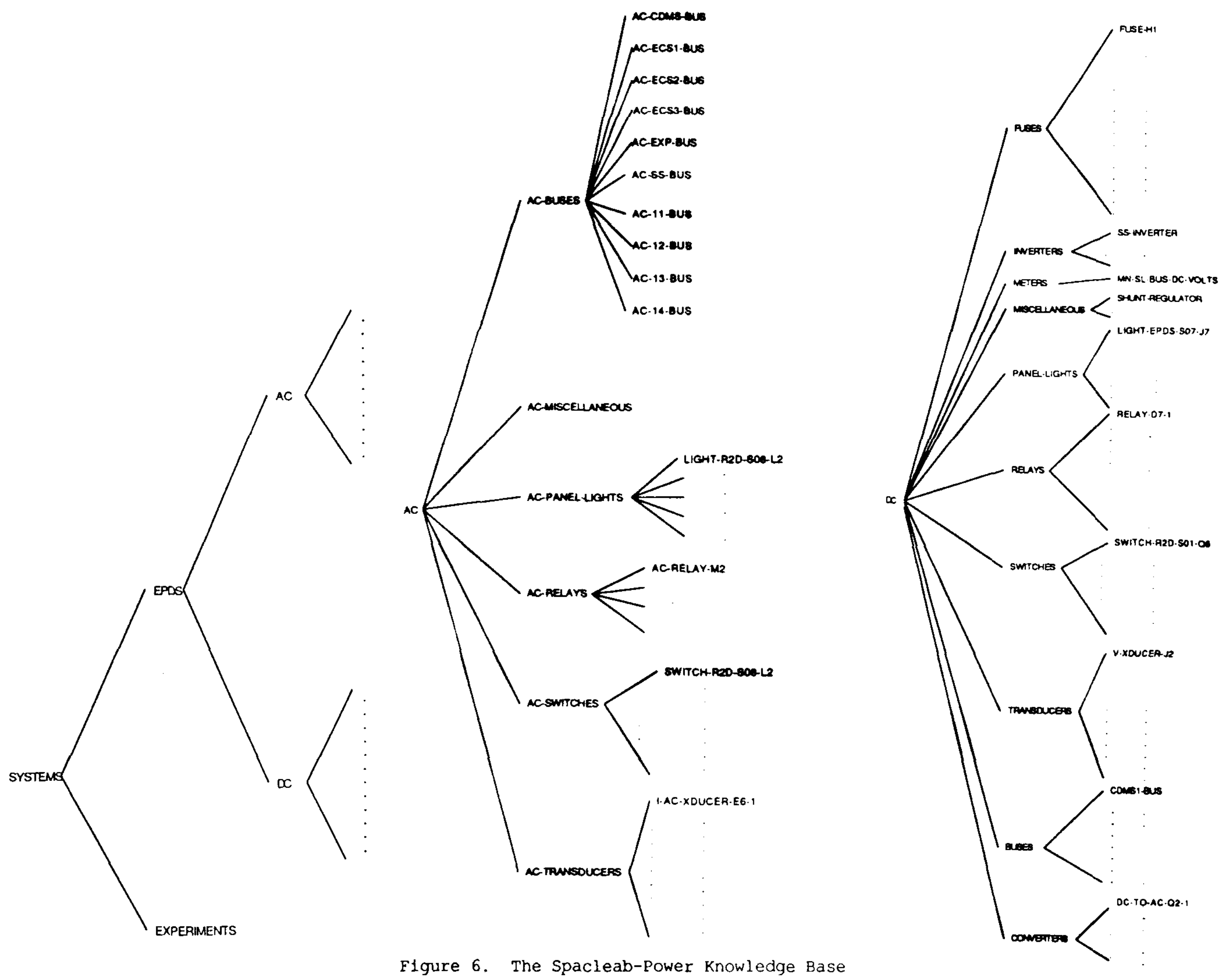


Figure 6. The Spaceleab-Power Knowledge Base

on the DC Distribution drawing and the AC distribution drawing of the Spacelab Systems Handbook. It is a perfect copy of these diagrams except for: the diodes are not represented and the return lines are not modeled. The Spacelab-Power KB contains a taxonomy of the objects to model the Spacelab Electrical System, including: Relays, Fuses, Switches, Lights, Meters, Buses, DC-to-AC Converters, AC-to-DC Converters, Inverters, and Transducers. Each electrical element is defined by the actions it can perform, its possible status, its current status, its possible connection, its current connections, and its drawing location. The actions it can perform are methods to which other elements or parameters send messages. This generates an object oriented environment where any element can send an electrical command to any other element without knowing anything about it. Usually the current connections define a message path to propagate the effects. Input parameters are attached to elements. These elements (objects) utilize KEE's Active Value mechanism as a watch-dog for detecting the out-of-limit value. When an out-of-limit value is detected, the appropriate malfunction procedure is activated. Figure 6 shows the taxonomy of the electrical elements of the Spacelab-Power KB.

The Procedures KB is used to store the diagnostic text that is displayed in the Malfunction Window and the SSR procedure that is displayed in the SSR Window. A procedure unit is composed of an active slot, a text slot, and several section slots. The active slot is used by the malfunction procedure kick-off mechanism to determine whether the malfunction has already been fired or not. The text slot, splices the choirs of text that the malfunction analysis needs after every partial conclusion. The section slots contain the malfunction procedure's partial conclusion text. Every section slot corresponds to a node in the malfunction procedure analysis tree (Figure 7) defined by the malfunction procedure structures. The malfunction procedure structure is itself a frame structure.

Display/User Interface Layer

The Display/User Interface Layer uses the knowledge available in the Graphics KB, Procedures KB, and the Spacelab- Power KB to handle all the displaying tasks in the user- interface. We will describe the Windowing and Menu system first.

(i) SLED Window-Frame

The windows and menus used in the *SLED* system are implemented through TI's extension of Common LISP with Window Flavors. Figure 8 shows the main system window. It is divided into six major sub-windows. The two windows in the upper left hand corner are the system time clock, labeled JULIAN, and the mission elapsed time clock, labeled MET. These times are used by the system for reference against timelines and schedules that affect the diagnostics that it performs. In addition, they are used to time-stamp the automatic log entries that the system makes. The time clock also controls the synchronized data acquisition cycle, in the Data Acquisition Layer.

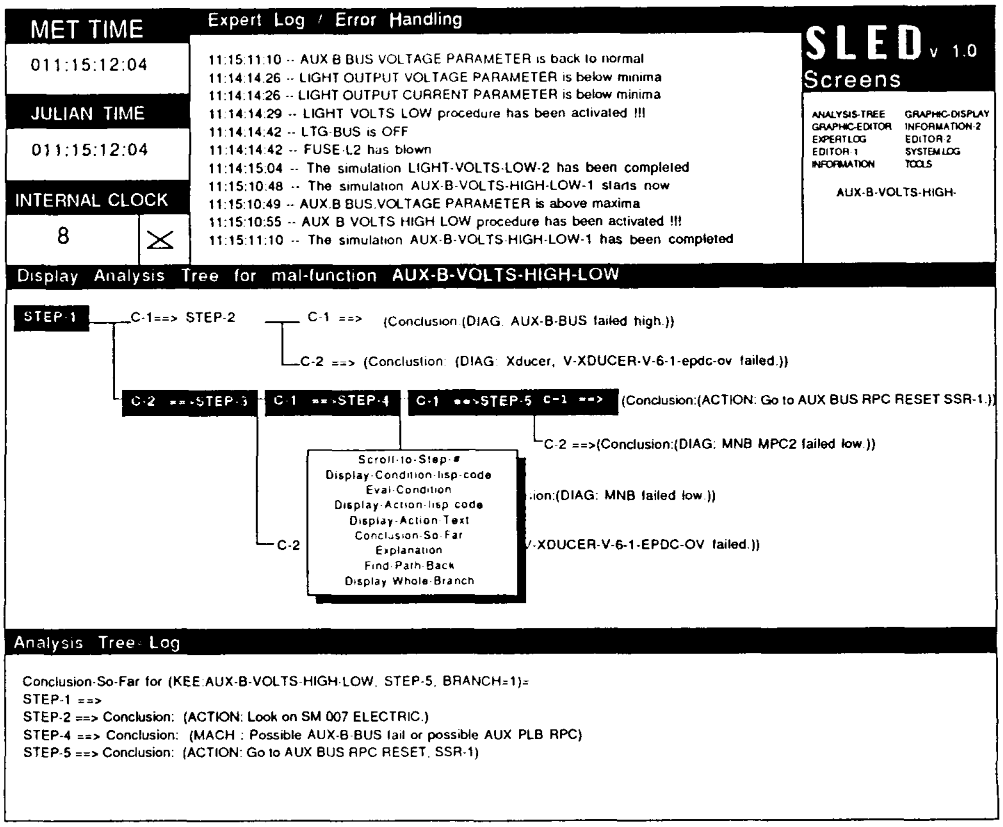
The log of information pertaining to the onboard electrical systems and user actions is located in the top center portion of Figure 8 and is labeled "Expert Log". This log displays the latest information about the status of the electrical systems on board. Each entry is time-stamped and a file containing a complete set of all entries is built for reference to previous events. The log is capable of displaying the last ten messages at all times.

The window at the right top corner of Figure 8, labeled "Screens", is a menu of the various screen configurations available for display. When malfunctions occur, a new entry will appear in this window and can be moused on to be displayed. Among these entries are "Graphic-Editor", "Tools", "Information", "Analysis-Tree", and "Graphic-Display". We will explain some of these screen later.

The thin window approximately one third of the way to the bottom of Figure 8, labeled "User Input", is used for the system to inquire information from the user. At times, all of the information required to make an accurate determination of a problem's cause is not contained in the telemetry downlink. In most instances such as this, a "yes" or "no" query will appear in this window. A positive response is made by clicking the mouse on the query. Not clicking on the query, within a specific period of time, communicates a negative response. At other times, the user may be asked to perform certain functions. A mouse click on the request is taken as confirmation that the action has been taken.

The bottom portion of the screen is the place for displaying different window configurations, selectable from the "Screens" window mentioned above. When the "Tools" entry in the "Screens" window is mouse-selected, or when the system is newly started up, the "Tools" window is displayed, (Figure 8). The "Tools" window is for accessing the set of utilities supplied for the manipulation of the system. Tools are supplied to modify the diagnostic algorithms, display various information about the status of electrical system hardware, display graphic representations of the electrical systems model, build test data sets, display logged information, change the systems status of various electrical system hardware, display SSR procedures, input the text for procedures, display graphic representation of the malfunction diagnostic trees, and display a trace of what caused a certain malfunction conclusion to be reached.

Each configuration has a set of functions associated with it. For instance, the Graphic-Editor configuration is used to build or edit the graphic representation of the electrical system model. When it is the current configuration, the graphic representation of the electrical systems can be modified, as can the electrical system model that the expert

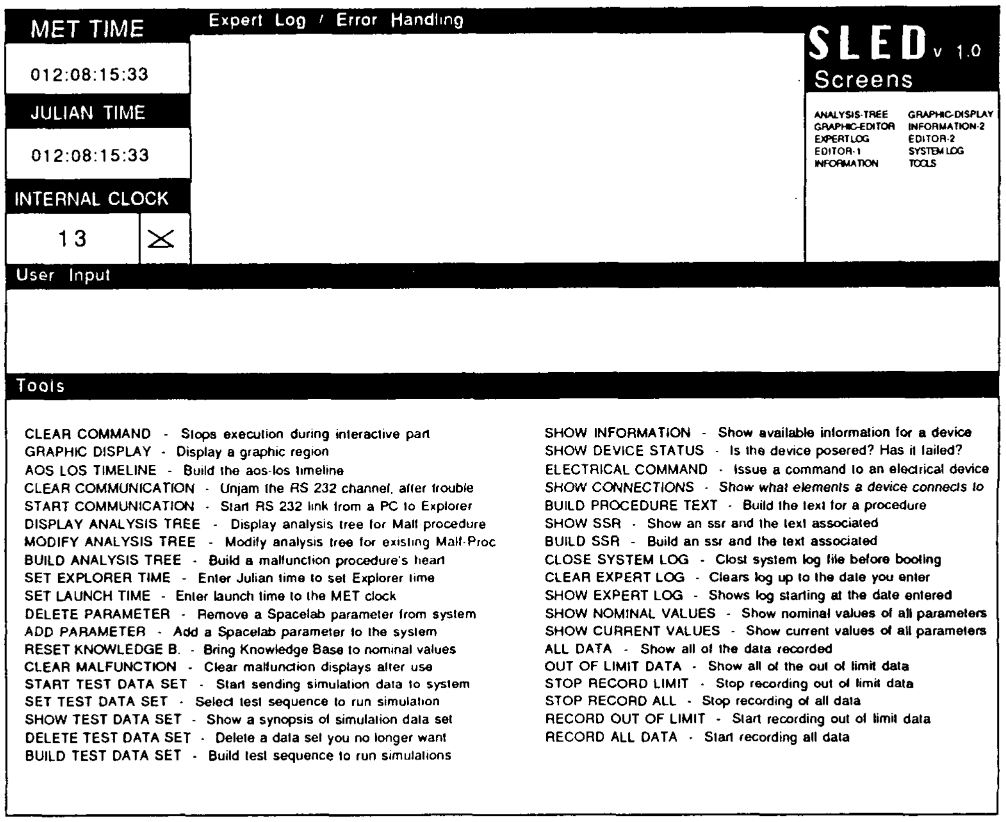


```
11:15:11:10 -- AUX B BUS VOLTAGE PARAMETER is back to normal
11:14:14.26 -- LIGHT OUTPUT VOLTAGE PARAMETER is below minima
11:14:14:26 -- LIGHT OUTPUT CURRENT PARAMETER is below minima
11:14:14:29 -- LIGHT VOLTS LOW procedure has been activated !!!
11:14:14:42 -- LTG-BUS is OFF
11:14:14:42 -- FUSE-L2 has blown
11:14:15:04 -- The simulation LIGHT-VOLTS-LOW-2 has been completed
11:15:10:48 -- The simulation AUX-B-VOLTS-HIGH-LOW-1 starts now
11:15:10:49 -- AUX.B BUS.VOLTAGE PARAMETER is above maxima
11:15:10:55 -- AUX B VOLTS HIGH LOW procedure has been activated !!!
11:15:11:10 -- The simulation AUX-B-VOLTS-HIGH-LOW-1 has been completed
```

```
STEP-1 ___C-1==> STEP-2   ___ C-1 ==>  (Conclusion (DIAG. AUX-B-BUS failed high.))
                              C-2 ==> (Conclustion: (DIAG: Xducer, V-XDUCER-V-6-1-epdc-ov failed.))
       C-2 ==>STEP-3  C-1 ==>STEP-4  C-1 ==>STEP-5 C-1 ==> (Conclusion:(ACTION: Go to AUX BUS RPC RESET SSR-1.))
                                                   C-2 ==>(Conclusion:(DIAG: MNB MPC2 failed low.))
                                                  ion:(DIAG: MNB failed low.))
                      C-2                         V-XDUCER-V-6-1-EPDC-OV failed.))
```

```
Conclusion-So-Far for (KEE:AUX-B-VOLTS-HIGH-LOW, STEP-5, BRANCH=1)=
STEP-1 ==>
STEP-2 ==> Conclusion: (ACTION: Look on SM 007 ELECTRIC.)
STEP-4 ==> Conclusion: (MACH : Possible AUX-B BUS fail or possible AUX PLB RPC)
STEP-5 ==> Conclusion: (ACTION: Go to AUX BUS RPC RESET. SSR-1)
```

Figure 7. Diagnostic Analysis Tree Display.

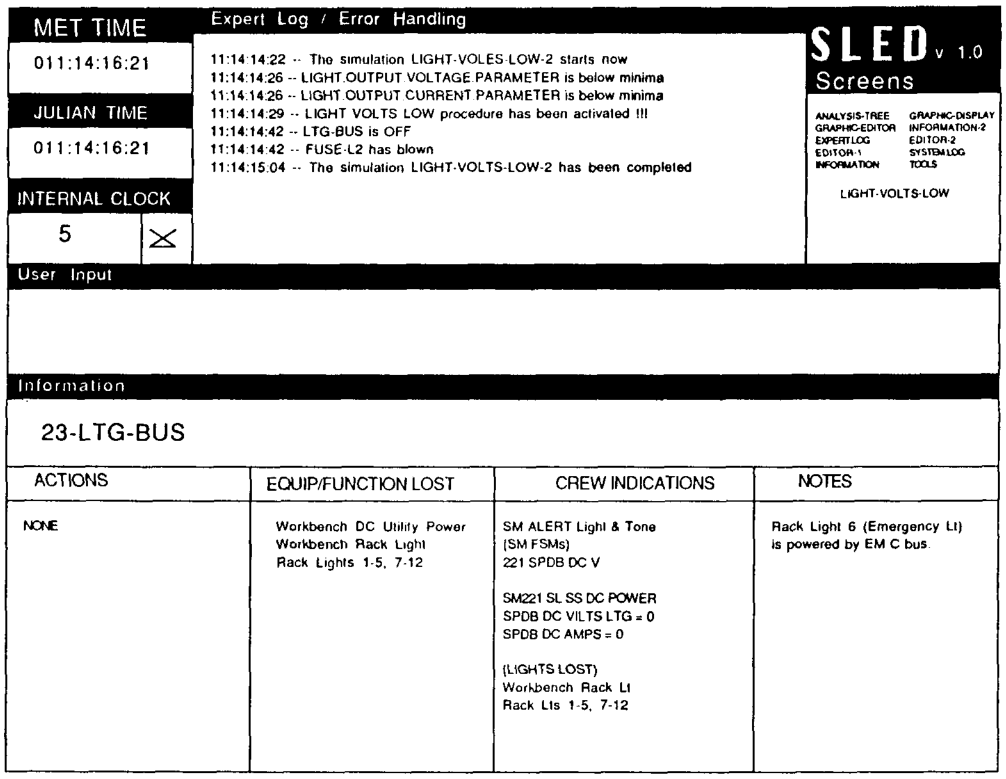


CLEAR COMMAND - Stops execution during interactive part	SHOW INFORMATION - Show available information for a device
GRAPHIC DISPLAY - Display a graphic region	SHOW DEVICE STATUS - Is the device posered? Has it failed?
AOS LOS TIMELINE - Build the aos-los timeline	ELECTRICAL COMMAND - Issue a command to an electrical device
CLEAR COMMUNICATION - Unjam the RS 232 channel, after trouble	SHOW CONNECTIONS - *Show what elements a device connects to*
START COMMUNICATION - Start RS 232 link from a PC to Explorer	BUILD PROCEDURE TEXT - Build the text for a procedure
DISPLAY ANALYSIS TREE - Display analysis tree for Malf procedure	SHOW SSR - Show an ssr and the text associated
MODIFY ANALYSIS TREE - Modify analysis tree for existing Malf-Proc	BUILD SSR - Build an ssr and the text associated
BUILD ANALYSIS TREE - Build a malfunction procedure's heart	CLOSE SYSTEM LOG - Clost system log file before booting
SET EXPLORER TIME - Enter Julian time to set Explorer time	CLEAR EXPERT LOG - Clears log up to the date you enter
SET LAUNCH TIME - Enter launch time to the MET clock	SHOW EXPERT LOG - Shows log starting at the date entered
DELETE PARAMETER - Remove a Spacelab parameter from system	SHOW NOMINAL VALUES - Show nominal values of all parameters
ADD PARAMETER - Add a Spacelab parameter to the system	SHOW CURRENT VALUES - Show current values of all parameters
RESET KNOWLEDGE B. - Bring Knowledge Base to nominal values	ALL DATA - Show all of the data recorded
CLEAR MALFUNCTION - Clear malfunction displays after use	OUT OF LIMIT DATA - Show all of the out of limit data
START TEST DATA SET - Start sending simulation data to system	STOP RECORD LIMIT - Stop recording out of limit data
SET TEST DATA SET - Select test sequence to run simulation	STOP RECORD ALL - Stop recording of all data
SHOW TEST DATA SET - Show a synopsis of simulation data set	RECORD OUT OF LIMIT - Start recording out of limit data
DELETE TEST DATA SET - Delete a data set you no longer want	RECORD ALL DATA - Start recording all data
BUILD TEST DATA SET - Build test sequence to run simulations	

Figure 8. SLED Main Window with the Tools Screen Displayed

ORIGINAL PAGE IS OF POOR QUALITY

ORIGINAL PAGE IS
OF POOR QUALITY



MET TIME
011:14:16:21
JULIAN TIME
011:14:16:21
INTERNAL CLOCK
5

Expert Log / Error Handling

```
11:14:14:22 -- The simulation LIGHT-VOLES-LOW-2 starts now
11:14:14:26 -- LIGHT OUTPUT VOLTAGE PARAMETER is below minima
11:14:14:26 -- LIGHT OUTPUT CURRENT PARAMETER is below minima
11:14:14:29 -- LIGHT VOLTS LOW procedure has been activated !!!
11:14:14:42 -- LTG-BUS is OFF
11:14:14:42 -- FUSE-L2 has blown
11:14:15:04 -- The simulation LIGHT-VOLTS-LOW-2 has been completed
```

SLED v 1.0
Screens

ANALYSIS-TREE	GRAPHIC-DISPLAY
GRAPHIC-EDITOR	INFORMATION-2
EXPERTLOG	EDITOR-2
EDITOR-1	SYSTEM-LOG
INFORMATION	TOOLS

LIGHT-VOLTS-LOW

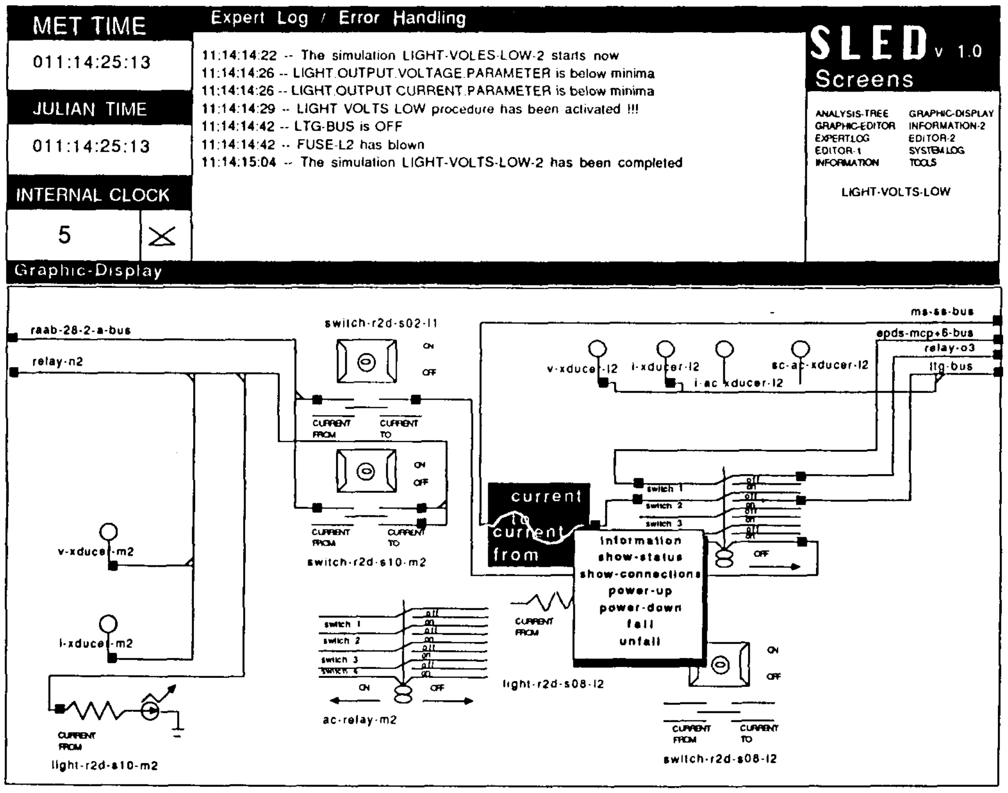
User Input

Information

23-LTG-BUS

ACTIONS	EQUIP/FUNCTION LOST	CREW INDICATIONS	NOTES
NONE	Workbench DC Utility Power Workbench Rack Light Rack Lights 1-5, 7-12	SM ALERT Light & Tone (SM FSMs) 221 SPDB DC V SM221 SL SS DC POWER SPDB DC VILTS LTG = 0 SPDB DC AMPS = 0 (LIGHTS LOST) Workbench Rack Lt Rack Lts 1-5, 7-12	Rack Light 6 (Emergency Lt) is powered by EM C bus.

Figure 9. SSR Procedure Display



MET TIME
011:14:25:13
JULIAN TIME
011:14:25:13
INTERNAL CLOCK
5

Expert Log / Error Handling

```
11:14:14:22 -- The simulation LIGHT-VOLES-LOW-2 starts now
11:14:14:26 -- LIGHT OUTPUT VOLTAGE PARAMETER is below minima
11:14:14:26 -- LIGHT OUTPUT CURRENT PARAMETER is below minima
11:14:14:29 -- LIGHT VOLTS LOW procedure has been activated !!!
11:14:14:42 -- LTG-BUS is OFF
11:14:14:42 -- FUSE-L2 has blown
11:14:15:04 -- The simulation LIGHT-VOLTS-LOW-2 has been completed
```

SLED v 1.0
Screens

ANALYSIS-TREE	GRAPHIC-DISPLAY
GRAPHIC-EDITOR	INFORMATION-2
EXPERTLOG	EDITOR-2
EDITOR-1	SYSTEM-LOG
INFORMATION	TOOLS

LIGHT-VOLTS-LOW

Graphic-Display

Figure 10. Graphic Display of Schematics

system uses to perform its diagnosis. The functions associated with the different mouse buttons are displayed at the bottom of the Explorer screen. The mouse buttons change from configuration to configuration but certain conventions have been adhered to wherever possible to make the operation of the system as consistent as possible.

The "Information" screen, "Graphic-Display" screen and the "Analysis-Tree" screen are used as buffers (they are also used for other purposes) to display malfunction-related information. When a malfunction window is visible (Figure 2) and a mousable item is clicked on, then these screens listed above are utilized. If the mousable item is an SSR item, then the "Information" screen is used to display the SSR-procedure, (Figure 9). If the mousable item is an electrical component then the "Graphic-Display" screen is used to display the graphic region, (Figure 10). If the mousable item is the "WHY ?", then the "Analysis-Tree" screen is used to display the diagnosis tree with the diagnosis path highlighted, (Figure 7). Each of these screens are also mouse- sensitive and supports further information querying about each electrical component and each diagnosis step. The mouse click is monitored by the main SLED-Window-Frame process for the window configuration control and the handling of mouse-sensitive items. Three subprocess: Display-Information-Process, Display-Graphics-Process, and Display-Tree-Process are used to display the different type of information (text, schematics, and diagnosis tree) into the appropriate windows.

(ii) Graphical Editor

In order to support the initial input and reconfiguration update of the schematics, a graphical editor is supported.

The graphical editor is one of the most complex features of the system. It is designed to make input of schematic drawings of the Spacelab electrical system a manageable task. It not only makes input of the drawings easier but also makes cross checks to insure that the graphics are accurate representations of the model used by *SLED* in its diagnostics. The graphics for SLS-1 have already been defined and input to the system.

The graphical editor is invoked by moving the mouse into the screens window and clicking on "Graphic-Editor". There are command regions defined by the boxes at the top of the graphic-editor window. Explanations of the uses of some of the commands follow:

choose-an-item: This is used to draw an image of the item within the region specified earlier. The devices that belong in each region have been predetermined by the region specified with which they have been named. Only devices that have been defined in the SPACELAB-POWER knowledge base will be available for graphic representation.

choose-a-drawn-item: This function is used to change the positions of items that have already been drawn. When this function is invoked by a left mouse click, a menu will appear giving as choices the item that have been previously drawn using the choose-an-item function. A left click on one of the items will cause that item to be erased from the position where it was drawn and redrawn where the mouse is when a middle click follows.

connections: The software provided for making the connections between items that are drawn in a graphic region are quite extensive. When invoked, the connections function will display a menu of the items drawn as choices for making connections to or from the item. Choosing an item will cause a menu to pop-up consisting of possible connection points for that item. This list of choices is build from the connections that are defined in the Spacelab-Power knowledge base. In addition to the possible connection points, choices also exist for adding, modifying, or deleting connections in the Spacelab-Power knowledge base. When a connection is made in the system, a box will appear at the connection points of the two items involved. To complete the connection image with wiring, left click where a wire is to terminate at one end. The next middle click will cause a wire to be drawn to the position of the mouse when the middle click is made.

(iii) Build-Analysis-Tree Tool

There are tools for building, modifying, and displaying the malfunction procedure analysis tree to support the reconfiguration capability. The "build-analysis-tree" tool allows the user to create his own analysis procedure. This tool is prompt and menu driven to make it easy for the user to build the malfunction procedure. The system first prompts the user for the malfunction name, comments, time delay, transient spike condition, notification message text, and number of steps involved in the diagnosis procedure. Then a session of menu prompts for conditions and actions to be performed in each diagnostic step are presented. Parameters to be monitored within this malfunction procedure are then solicited by the system with all available parameters being displayed and will then let the user make choices and specify the upper and lower limit condition. Logical and numerical operators are available from the menus to create complicated conditions for testing in each diagnostic step. The action part of the diagnostic step is then requested by the system with menu prompts allowing the user to make choices from those menus. Among those possible actions include: changing the status of any electrical element by sending a message, query the user for input, schedule a task to run in some specific time from the current time, build the text for the current diagnostic step to be displayed in the malfunction window, set the next diagnostic step, and stop analysis, which tells the system that a conclusion has been reached. After the user completes the build analysis tree session, the malfunction procedure just built will be displayed as a diagnostic tree and the user will have the option of modifying it or saving and defining it to the system. If the user selects the define/save option, the newly built malfunction procedure will be activated if the input parameter violates the value range specified by the user in the build-analysis-tree session. That is, the malfunction procedure and the monitoring mechanism is automatically set up in the build-analysis-tree tool.

(iv) Other Tools

There are other useful tools available in SLED. It provides a log for the status of the out-of-limit conditions along with a date and time stamp. The user may look at this log at any time. The log may also be printed, since all of the logged entries are saved to a file. The user can view the current and nominal values of the parameters that the system looks at. He can obtain the status of a particular electrical component. He may also see what other components are connected to it along with specific information about the component. One of the major capabilities of this system is that it allows the user, a non-programmer, to create a malfunction recovery procedure, should the need arise. The different tools that allow the user to do this include building the diagnostic tree (described above), building the procedure text, the text that is displayed in the unique malfunction recovery window, building the SSR text, the text that is viewed for the recovery procedure, and adding the necessary parameters needed to diagnose the malfunction. The user also has the capability of deleting, modifying, or viewing the different parts of the malfunction recovery procedure. This includes modifying or viewing the malfunction recovery procedure, viewing the SSR text, and deleting or viewing the parameters. The user may also set up an AOS-LOS timeline. This is used to "wake-up" the system during AOS, and put the system to "sleep" during LOS. In order to test the *SLED* system, tools were incorporated into the system to test its reliability. These tools includes building, viewing, deleting, and setting the test data sets.

In addition to these tools, specifically designed for *SLED*, the standard user-interface tools of KEE are always available for the advanced user in data monitoring or KB updates. For example, we can use the KEE user-interface for parameter monitoring and trend analysis.

4. CURRENT STATUS

Testing

The validation process at the current time is not formal. Three levels of testing are planned to be performed: 1. Hand- crafted simulation data, within the Explorer; 2. POCC simulator or tape of real data with Explorer to VAX communications. 3. Real-Time operational test during the SLS-1 mission. We have used hand-crafted test-data stored as KEE's units and user interaction to aid testing. We plan to use the POCC simulator and taped records of real data to test the system. Due to the complexity and dynamic nature of the problem domain, the real system must be verified by actual on-line testing during the SLS-1 mission. We plan to have the *SLED* system in operation, in parallel to the conventional system during the SLS- 1 mission scheduled for June 1990.

During the SLS-1 mission, it is planned to support the *SLED* system in parallel with the current screen displays and monitoring tools of the POCC operation facility. The *SLED* terminal will be located next the the existing POCC electrical power monitoring terminal. This arrangement will also show the capability of *SLED* to the PSE and enhance the user acceptance of the *SLED* expert system.

While the testing of *SLED* with the simulator and the final testing of the *SLED* system during the SLS-1 mission is still on schedule, evaluation by the domain expert so far has been positive. We will not address the problem of how to measure the performance of *SLED* as a trouble shooting expert system in a formal sense. Instead, the actual comparison of the *SLED* performance with the current monitoring system and the PSE's performance in diagnosis of the fault, is of great interest to us. The *SLED* system (fault) log file will be used as an audit trail to verify the overall system performance against the written fault logs produced by the PSE.

Currently we have finished level 1 testing procedures. The POCC simulation software is currently in development and will be available in August 1989. The hand-crafted test case includes data that fired multiple malfunctions. All the tests that were performed went well and the response time was about 15 seconds in handling a single malfunction. This time was measured from the detection of the out-of-limit value to the completion of the diagnosis of the malfunction, assuming immediate user action for all user input requested. It also takes 15 seconds for the 2 malfunction case, and it takes about 30 to 50 seconds for a 5 malfunction case. The point here is that the time needed for multiple malfunctions is growing linear with the number of malfunctions. In particular it is not increasing exponentially with the number of malfunctions occurring. The reason for this linearity of response time is the way we treat each malfunction independently in a separate window with an independent diagnostic process. No interrelations between the malfunctions are considered in our model. Therefore, the diagnosis process of each malfunction is running in parallel. In theory, the response time of the multiple malfunction case should be about the same as the single malfunction case. But the fact that independent processes are time-sharing processes in the host LISP machine and that there is only one user interface window, makes the total time grow approximately linear with the number of malfunctions occurring.

Enhancement Plan

As mentioned in the Background section, the current *SLED* system is the first phase of the Spacelab electrical power system which contains only the knowledge about the EPDS. No knowledge of the Experiment equipment have been included. We are currently adding the malfunction procedures, the SSR's, and the schematics of the subsystem up to the experiment hardware to the *SLED* system, as required in phase 2, and then, finally adding the knowledge of each experiment equipment to the system (phase 3). Obviously, a lot more knowledge needs to be added to the system. This is making us approach the limitation of the available hardware and software. The total number of frames, in KEE it is referred to as units, is 363 units within the Spacelab- Power KB, 519 units in the Test-Data KB, 125 units in the Graphics KB, and 83 units in the Procedures KB. The total number of experiment equipments that we would like to represent is 40. For each experiment we need to increase the Graphics KB by approximately 10 units and the Spacelab-Power KB by approximately 50 units. Besides, there is a lot of disk space that is used to store the bit-maps for the schematics, 4.7 MB of disk space is used for the schematics of the Spacelab EPDS- distribution. It becomes impractical to use the current scheme, i.e. direct bit-map storage to store all the schematics for the experiment equipment to be monitored. A better way of storing the schematics needs to be implemented. The new approach we have in mind is to build up the schematics on-the-fly when requested. The schematics will be built using the electrical component and connection information in the KB's. Of course, there is a trade-off between time and disk and memory space. The on-the-fly approach will slow down the response time for the schematics display. Fortunately, the schematic display is not as time-critical as the monitoring, diagnostic, and malfunction procedure portion. The tremendous disk space saving justifies the slow response time in the schematic display.

The current system is hosted in the Texas Instruments (TI's) Explorer Lisp-Machine. Our target delivery system will be TI's Micro-Explorer, i.e. a Mac II with TI's Lisp processor board. We will start the conversion when our Micro-Explorer system is available, which will be in March of 1989. We expect the conversion effect to be a minimal.

We would like to make the knowledge acquisition tools: the graphical editor, used for creating and updating the schematics, and the diagnostic tree building tools, used to input and update the malfunction procedures, to be more user friendly in the near future, which will incorporate the users suggestions.

We would also like to use rule representations of the diagnosis logic instead of the current explicit representation of the logic. This would be a major overhaul and the task of meeting the real time speed requirements may pose some tough problems, because KEE's rule system is slow.

5. CONCLUSION

We have successfully demonstrated the applicability of expert system technology to the Spacelab power subsystem diagnostic problem. The response time of about 15 seconds, i.e. the time needed for 5 data cycles, for a malfunction diagnosis, with the corrective malfunction procedure, SSR procedure and schematics readily available, well exceeds the performance of a trained PSE. Besides the on-line storage and fast retrieval of the schematics and the availability of status information for each electrical component will make the system a big help for the PSE in dealing with the malfunctions which are not covered in the malfunctions procedure handbook.

With our experience of the development of the *SLED* system, there are several important points in real time expert system development we would like to re-iterate:

1. To narrow down the application domain is a very important design issue. After actually trying to solve a more general problem, the *SLED* system evolved from the General Diagnostic System. This point has been emphasized so many times in the literature, e.g. [Hayes-Roth, Waterman & Lenat, 1983], it is too easy to overlook.

2. Expert systems for real time operations is achievable. Besides the planned enhancements, the system can be extended to other areas of spacecraft mission control problems.

3. The first use of *SLED* was for training. As has been pointed out in the literature, e.g. [Buchanan & Shortliffe, 1984], the explanation capability makes expert systems an excellent tool for training.

4. The system can be used as a validation and verification tool for the user to generate the malfunction procedures. It can be extended to a tool for generating the malfunction procedure document.

5. When the next generation of Lisp-machines based on a Lisp chip is stabilized, it will be feasible to put the Lisp-based fault diagnostic expert system on-board Spacelab. Actually, this approach would be simpler and may speed up the fault diagnostic task of the experiment, because we can further partition the problem and place the expert system hardware and software directly in the relevant area. For example, the *SLED* system can be placed inside the experiments box and does not need to store the system model and knowledge about Spacelab itself.

Acknowledgement

We would like to thank Marshal Space Flight Center (MSFC), Johnson Space Center (JSC), and the Electrical group of the Systems Engineering section of GE Government Services Houston for their support and guidance during this project.

References

[JSC-18927]
Flight Data File, Spacelab Malfunction Procedures, Basic, Revision B, March 1985, Mission Operations Directorate, JSC, NASA.

[JSC-12777C]
Spacelab Systems Handbook, Rev. C, Aug. 1983, Mission Operations Directorate, JSC, NASA.

[LS-50044-A]
Data Requirements for SLS-1 JSC Life Sciences Flight Experiment, pp. 2.2 - 2.5.

[SLP/2104]
Spacelab Payload Accommodation Handbook, Main Volume, issue no: 2, revision: 0, Aug. 1985, J. W. Thomas Manager, NASA MSFC Spacelab Program Office.

[Buchanan & Shortliffe, 1984]
Rule-Based Expert Systems; B. Buchanan, and E. Shortliffe; Addison-Wesley, Reading, MA., 1984.

[Cantone, Pipitone & etc., 1983]
Model-Based Probabilistic Reasoning for Electronic Trouble- Shooting; R.R. Cantone, F.J. Pipitone, W.B. Lander, M.P. Marrone; Preceedings IJCAI 1983.

[D'Ambrosio, Fehling & etc., 1987]
Real-Time Process Management for Materials Composition in Chemical Manufacturing; B. D'Ambrosio, M.R. Fehling, S. Forrest, P. Ranlef, and B. Michael Wiber; IEEE/Expert, Summer 1987.

[Davis, 1983]
Reasoning From First Principles in Electronic Trouble-Shooting; Randall Davis; International Journal of Man-Machine Studies, Vol. 19, 1983.

[Dickey & Toussaint, 1984]
An Application of Expert Systems to Manned Space Stations; F.J. Dickey and A.L. Toussaint; CAIA 1984.

[Forgy, 1981]
OPS-5 User's Manual, Report CMU-CS-81-135; C.L. Forgy, Dept. of CS, Carnegie-Mellon University.

[Geffner & Pearl, 1987]
Distributed Diagnosis of Systems with Multiple Faults; H. Geffner and J. Pearl; CAIA 1987.

[Griesmer, Houg & etc., 1985]
YES/MVS: A Continuous Real-Time Expert System; J.H. Griesmer, S.J. Houg, M. Karnaugh, J.K. Kastner, M.I. Shor; IJCAI 1985.

[Hamilton, 1986]
SCARES-A Spacecraft Control Anomaly Resolution Expert System; M. Hamilton; Expert System in Government Symposium, 1986.

[Hayes-Roth, Waterman & Lenat, 1983]
Build Expert Systems; F. Hayes-Roth, D.A. Waterman, and D.B. Lenat; Addison-Wesley, Reading, MA, 1983.

[KEE 2.1]
IntelliCorp KEE Software Development System User's Manual; July 18, 1985.

[Leinweber, 1987]
Expert Systems in Space; D. Leinweber; IEEE/Expert, Spring 1987.

[Maletz, 1985]
An Architecture for Consideration of Multiple Faults; M.C. Maletz; CAIA 1985.

[Milne, 1987]
Fault Diagnosis Through Responsibility; R. Milne; IJCAI 1987.

[Muratore, Madision & etc., 1988]
Real Time Expert System Prototype for Shuttle Mission Control; J.F. Muratore, R.M. Madison, T.A. Heindel, T.B. Murphy, R.F. McFarland, A.N. Rasmussen, E.D. Kindred, C.L. Whitaker; Telemetry Conference, 1988.

[Pipitone, 1984]
An Expert System for Electronic Trouble-Shooting Based on Function and Activity; F. Pipitone; CAIA 1984.

[Rook & Odubiyi, 1986]
An Expert System for Satellite Orbit Control (ESSOC); F.W. Rook and J.B. Odubiyi; Expert Systems in Government Symposium, 1986.

[Silverman, 1986]
Facility Advisor: A Distributed Expert System Testbed for Spacecraft Ground Facilities; Barry G. Silverman; Expert Systems in Government Symposium, 1986.

[Stallman & Sussman, 1979]
Problem Solving about Electrical Circuits; R.M. Stallman and G.I. Sussman; In Artificial Intelligence: An MIT Perspective, 1979, Edited by P. Winston and R. Brown.

[Strandberg, Abramovich, & etc., 1985]
PAGE-1: A Trouble-Shooting Aid for Non impact Page Printing Systems; C. Strandberg, I. Abramovich, D. Mitchell, K. Prill; CAIA, 1985.

[Taie & Srihari, 1986]
Device Modeling for Fault Diagnosis; M.R. Taie and S. Srihari; Expert Systems in Government Symposium, 1986.

[Wilkinson, Happell, & etc., 1988]
Achieving Real-Time Performance in FIESTA; W. Wilkinson, N. Happell, S. Miksell, and R. Quillin; 1988 Goddard Conference on Space Application of Artificial Intelligence, Goddard Space Flight Center, NASA, May 1988.